To design a Url shortener service of a big scale a good idea would be to use multiple servers for different continents, for minimum latency. It would be also useful to add cache for often accessed links. Creation of unique Url should be executed beforehand, so when user generates link - we give him existing url. This way the request processes faster, than it would otherwise. We can use numbers and upper and lowecase letters to generate url. Assume, url length is 7, than we have more than 3500 billion combinations. We can use counter to keep track of already generated links. Than we change counter to base62 – and that is how we get url. When counter reaches maximum - just set it to 0.

Let's assume we have 10,000 generate requests and a record approximately takes up to 2000 bytes(including all fields) => 1.7 TB a day. Each day more and more urls will be generated. So, we need to use database, which can scale easily and also perform fast read and write. That is why I would use one of NoSQL dbs, not MySql, which was used in the prorotype. One of possible solutions is MongoDB. It stores data in json-like format, so it would be easy to store documents (shortUrl, originalUrl, createDate, updateDate, clicks etc). Each day takes up ~860 million of url combinations, which means we need to delete some records in under 4069 days. But we also need to remember about limited size of the database. Although it would be better to keep even unclicked links alive for at least 2 months, if we have limited space, we can delete least clicked links every 2 days (but will it be a good user experience, that is the question).